Privacy-Preserving Speech Processing via STPC and TEEs

(Extended Abstract)[*]

Sebastian P. Bayerl[1], Ferdinand Brasser[2], Christoph Busch[3], Tommaso Frassetto[2], Patrick Jauernig[2], Jascha Kolberg[3], Andreas Nautsch[3,4], Korbinian Riedhammer[1], Ahmad-Reza Sadeghi[2], Thomas Schneider[2], Emmanuel Stapf[2], Amos Treiber[2], and Christian Weinert[2]
[1]TH Nürnberg, Germany
{sebastian.bayerl,korbinian.riedhammer}@th-nuernberg.de
[2]TU Darmstadt, Germany
{ferdinand.brasser,tommaso.frassetto,patrick.jauernig,ahmad.sadeghi,emmanuel.stapf}@trust.tu-darmstadt.de
{schneider,treiber,weinert}@encrypto.cs.tu-darmstadt.de
[3]Hochschule Darmstadt, Germany
{busch,kolberg}@h-da.de
[4]EURECOM, France
andreas.nautsch@eurecom.fr

ABSTRACT

With the advent of mobile and smart-home devices such as Amazon Alexa or the Google Assistant providing voice-based interfaces, voice data is commonly transferred to corresponding cloud services. This is necessary to quickly and accurately perform tasks like automatic speaker verification (ASV) and speech recognition (ASR) that heavily rely on machine learning.

While enabling intriguing new applications, this development also poses significant risks: Voice data is highly sensitive since it contains biometric information of the speaker as well as the spoken words. Thus, the security and privacy of billions of end-users is at stake if voice data is not protected properly. When developing privacy-preserving solutions to mitigate such risks, it is also important to keep in mind that the involved machine learning models represent intellectual property of the service providers and therefore must not be revealed to users.

The contribution of our work is three-fold: First, we present an efficient architecture for privacy-preserving ASV via outsourced secure two-party computation (STPC). Compared to existing solutions based on homomorphic encryption (HE), the verification process is 4,000x faster, while retaining a high verification accuracy and guaranteeing unlinkability, irreversibility, and renewability of stored biometric data.

Since cryptographic secure computation protocols currently do not scale to more involved tasks like ASR, we then present *VoiceGuard*, an architecture that efficiently protects speech processing inside a trusted execution environment (TEE). We provide a proof-of-concept implementation and evaluate it on speech recognition tasks isolated with Intel SGX, a widely available TEE implementation, demonstrating even real time processing capabilities.

Finally, we present *Offline Model Guard* (OMG) to enable privacy-preserving speech processing on the predominant mobile computing platform ARM even in offline scenarios. Beyond relying on the Intel SGX equivalent ARM TrustZone, we employ the security architecture SANCTUARY (NDSS'19) for strict hardware-enforced isolation from all other system components. Our prototype implementation performs privacy-preserving keyword recognition using TensorFlow Lite in real time.

1 INTRODUCTION

Devices providing voice-based interfaces are omnipresent in today's world. Amazon Alexa, Apple Siri, Google Assistant, or Microsoft Cortana are available to the more than two billion smartphone users in 2019. Also, there is a steadily increasing number of smart-home devices, like Amazon Echo, Apple HomePod, or Google Home, solely relying on voice-based interaction. Possible application scenarios are not restricted to the consumer market but increasingly cover professional activities, for example enterprise-ready smart assistants guiding through complicated business processes in order to increase productivity or banks remotely verifying users' identities via their biometric voice characteristics.

In any of the aforementioned cases, voice data is commonly transferred to the cloud for remote speech processing, such as automatic speaker verification (ASV) and speech recognition (ASR) that heavily rely on machine learning. This poses significant security and privacy risks since voice data contains sensitive biometric information as well as the spoken words: in case unprotected voice data gets out of hand, it may be abused, e.g., for impersonation attacks, assembling fake recordings, or simply extracting intimate as well as secret and sensitive content.

Privacy breaches in this domain are a reality: in 2018, a customer requested his recording archive from Amazon, but accidentally obtained access to 1,700 audio files from a stranger [23]. Furthermore, state authorities ordered Amazon to hand out recordings as they might contain evidence of crime [9].

A naive solution to these problems is to ship the speech processing code together with corresponding models to users to run locally. However, handing out such models in unencrypted form is mostly not in the interest of the service provider: A production-level model constitutes intellectual property since the underlying training data is usually hard to obtain and the creation of an accurate yet compact model requires extensive expertise. Furthermore, if attackers have unrestricted model access, the privacy of people represented

[*]This extended abstract combines three works which were recently published [5] or are currently in submission [3, 26].

in the training data is even more threatened by, e.g., membership inference attacks [25] and unintended memorization [7].

Solutions based on cryptography, i.e., homomorphic encryption (HE) or secure multi-party computation (SMPC), guarantee that neither user nor vendor need to reveal their respective input data in the clear. For ASV, we present an efficient privacy-preserving architecture based on outsourced secure two-party computation (STPC) [26]. However, secure computation protocols still infer a high overhead in computation time and communication costs. For more involved tasks like ASR, we therefore present *VoiceGuard* [5], the first architecture that efficiently protects speech processing inside a trusted execution environment (TEE). Furthermore, we present *Offline Model Guard* (OMG) [3] to enable privacy-preserving speech processing on the predominant mobile computing platform ARM even in offline scenarios.

2 SPEAKER VERIFICATION [26]

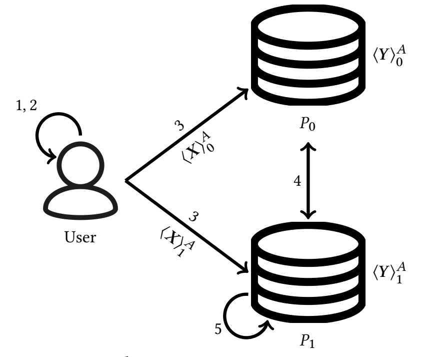
In ASV, the biometric characteristics of a speaker's voice are used to verify the speaker's identity. According to international standards for biometric information protection (BIP) [13], stored data needs to be *unlinkable*, *irreversible*, and *renewable*, meaning that the data from an individual cannot be linked across services, that no biometric data can be reconstructed from it, and that updates of the architecture without re-capturing biometric data are possible. Recently, Nautsch et al. [18] showed how to achieve BIP in ASV based on probabilistic embeddings (i-vectors) using HE. In addition, the vendor's model can also be protected. However, the usage of HE introduces a significant overhead, especially for the variant that protects the model. To reduce this overhead, we show how one can efficiently hide the model and enable BIP in embedding-based ASV using outsourced STPC [26].

Architecture Overview. Our architecture protects stored biometric information and the model used for the verification while making no additional assumptions compared to the architecture of [18]. The basic idea is to use the concept of outsourced STPC [14], where two servers P_0 and P_1 are in possession of *secret shares* of the user's data and the vendor's model and can therefore invoke STPC to securely compute the verification without gaining any knowledge about the inputs (assuming that both servers do not collude to exchange their shares and recover the secret).

In Fig. 1, our architecture is shown for the example of ASV based on the cosine score S_{cos} between the stored reference embedding $\boldsymbol{Y}$, the freshly extracted probe vector $\boldsymbol{X}$, and a threshold η. The user is authenticated if $S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) > \eta$. The secret shares $\langle Y \rangle_0$ and $\langle Y \rangle_1$ are generated by the user during enrolment.

To optimize this architecture for ASV, we employ a mix of different protocols that operate on different sharing types. The ABY framework [10] provides methods to efficiently convert between the different sharing types and protocols. Arithmetic shares in the GMW STPC protocol [12] allow for highly efficient score computation. After S_{cos} is securely computed using GMW, the corresponding arithmetic share is transformed into a share for Yao's garbled circuit protocol [29], which allows to efficiently compute the threshold comparison. Both resulting shares are then used to reveal the computed decision to the authenticating server P_1. We extend this basic mixed-protocol outsourced STPC architecture to more complex state-of-the-art ASV architectures relying on probabilistic linear discriminant analysis (PLDA) by also secretly sharing the model parameters between the servers and evaluating the PLDA-based ASV using the same mix of STPC protocols.

This architecture achieves unlinkability and irreversibility because of the information-theoretic security of the employed secret sharing. Due to the additively homomorphic property of the arithmetic secret sharing, it is also renewable without requiring the user to re-enroll. Our solution has the same requirements as [18], because it assumes that the servers honestly follow the protocol (semi-honest security) and that that they do not collude. The latter enables device-independent ASV because the user does not need to store a key. Moreover, our solution is secure against a malicious user, because the user would need to guess a correct probe in order to be verified successfully. This is not the case in [18], where a malicious user can be authenticated just by sending an encrypted, accepting score to the authenticating server.



1. User extracts probe $\boldsymbol{X}$.
2. User shares $\boldsymbol{X} = \langle \boldsymbol{X} \rangle_0^A + \langle \boldsymbol{X} \rangle_1^A$.
3. User sends $\langle \boldsymbol{X} \rangle_0^A$ to P_0 and $\langle \boldsymbol{X} \rangle_1^A$ to P_1.
4. Servers P_0 and P_1 securely compute:
   4.1. $\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) \rangle^A$
   4.2. $\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) \rangle^Y = A2Y(\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) \rangle^A)$
   4.3. $\langle \textbf{yes / no} \rangle^Y = \langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) > \eta \rangle^Y$
5. P_1 performs $Reconstruct(\langle \textbf{yes / no} \rangle^Y) \rightarrow \textbf{yes / no}$.

Figure 1: Privacy-preserving ASV based on comparing the cosine score between reference and probe embeddings Y and X to a threshold η using outsourced mixed-protocol STPC. $\langle X \rangle_i^T$ denotes the secret share of X held by party $P_{i \in \{0,1\}}$ of a type used for the arithmetic GMW STPC protocol ($T = A$) or for Yao's garbled circuit protocol ($T = Y$).

Evaluation. We implemented our architecture using the STPC framework ABY [10] and evaluated it on the NIST i-vector machine learning challenge [17] consisting of 1,306 reference identities and 9,634 probes, forming a progress and an evaluation set. We report on the results of the latter, which consists of 7,542,271 comparisons. The i-vectors are given with 5-digit precision. Our implementation supports full floating point operations for the score computation in the Boolean GMW protocol as well as a scaled score computation in the arithmetic GMW protocol, which can greatly improve efficiency by representing the embeddings as integers.

As for the biometric performance, an evaluation using error trade-off plots [16] shows that the scaled version of our architecture does not decrease verification accuracy in any way. Our scaling, however, limits the precision of the threshold η a service can provide to three digits. Nonetheless, for ASV and current technological demands, we assume that even 2-digit precision is sufficient. Our evaluation shows that even our scaled architecture achieves a recognition performance identical to the plaintext ASV.

To evaluate the runtime performance, we ran our implementation on two machines equipped with Intel Core i9-7960X CPUs and 128 GB of RAM connected with a bandwidth of 1 Gbit/s and 1 ms round trip time to emulate well-connected service providers. We compare our solution to the implementation of [18] that hides the subject data and the vendor's model. Though parts of the computation need to be computed on the user's device in [18], we ran the implementation on one server to obtain comparative benchmarks.

ABY allows to split the secure computation of a functionality into an input-independent offline phase and an input-dependent online phase. The offline phase can be executed at any time before the actual verification procedure and is therefore not considered relevant to the verification times. The results show that compared to the HE-based solution, our architecture significantly reduces ASV runtimes. For the cosine-based verification, our implementation takes between 3.7 ms and 4.2 ms for i-vector dimensions ranging from 50 to 600. This displays an improvement of 6x to 47x compared to [18]. The advantages of using outsourced STPC for ASV become clear when services want to use state-of-the-art ASV technologies that rely on PLDA. In that scenario, our implementation takes between 11 ms and 530 ms, which is an improvement of factor 1,421x to 4,099x with regards to [18]. Even when accounting for the offline phase, our architecture is still faster than the previous work. In that case the total evaluation time ranges from 191 ms to 19 s, the latter still being an improvement of 112x. Our experimental evaluation shows that STPC optimized for the use case of ASV is more efficient than HE, effectively reducing the HE verification times from about 36 minutes to about half a second.

3 VOICEGUARD [5]

VoiceGuard [5] is the first architecture that efficiently protects speech processing tasks by using a trusted execution environment (TEE), concretely the widely available Intel Software Guard Extensions (SGX). SGX allows the secure processing of confidential data even in a hostile environment by combining cryptographic techniques with hardware-enforced code and data isolation. Concretely, SGX introduces the concept of *enclaves*, which are programs executed in isolation from all other software on a system, including privileged software, like the operating system (OS) or a hypervisor.

VoiceGuard can easily be extended to enable user-specific models, such as feature transformations (including fMLLR), i-vectors, or model transformations (e.g., custom output layers).

Architecture Overview. For the sake of simplicity we explain our solution based on the ASR scenario visualized in Fig. 2. VoiceGuard works in three phases: (I) preparation, (II) initialization, and (III) operation. In the first phase, user U and vendor V need to agree on the code to be executed in the enclave.

In the second phase, the enclave code is instantiated. U and V use remote attestation (RA) to establish secure channels with the enclave through which they provision their respective encryption keys to the enclave. In particular, RA enables an external party to verify whether the enclave was created correctly, i.e., a cryptographic hash of the initial memory state of the enclave is signed by the platform signing key which is built into the CPU.

In the third phase, the enclave is ready to perform speech processing. Using the keys transmitted in the previous phase, U and V provide their respective inputs to the enclave in encrypted form. The user's input consists of audio samples and, if applicable, also user-specific adaptation parameters θ (e.g., i-vectors). The vendor's input consists of an acoustic model *AM*, typically a deep neural network (DNN), and a language model *LM*, typically a decoding graph. The result of the operation phase (which may contain updated adaptation parameters θ) is encrypted with the user's key, so only she can decrypt it. Once in the operation phase, the system can be queried repetitively by the user, thereby avoiding repeated preparation and initialization costs.

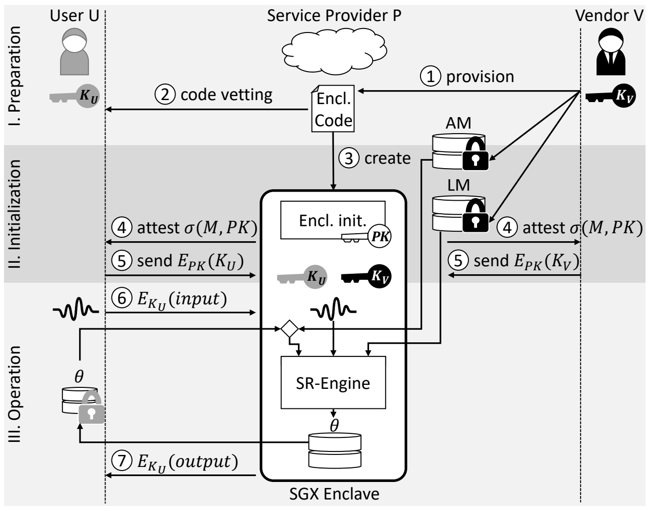


Figure 2: VoiceGuard architecture. User U establishes a secure channel with the Intel SGX enclave hosted at untrusted service provider P and sends sensitive voice data as well as user-specific adaptation data θ. Similarly, vendor V sends the sensitive acoustic and language models (*AM* and *LM*, respectively) through a secure channel. P securely processes U's voice data using V's models within an SGX enclave.

The protection guarantees of VoiceGuard hold even if the adversary is in full control over the software in the service provider's infrastructure (including privileged software like the OS or a hypervisor). However, we assume that the adversary cannot perform invasive hardware attacks like extracting keys from the CPU and we consider physical side-channel attacks, like differential power analysis [15], out of scope. To protect against side-channel attacks leveraging micro-architectural effects, the enclave developer has to incorporate appropriate defense mechanisms [4, 8, 24].

Evaluation. We created a proof-of-concept implementation which embeds the ASR toolkit kaldi [19] in an Intel SGX enclave using the Graphene library OS [27]. We ran experiments on two representative corpora: DARPA Resource Management (RM) [20] and Wall Street Journal (WSJ) [11]. For RM, the trained DNN is about 3 MB (9 hidden layers, 750 k parameters), the uni- and bigram decoding graphs are 0.5 MB and 2 MB, respectively. For WSJ, the trained DNN is about 14 MB (15 hidden layers, 3.6 M parameters), the pruned trigram decoding graph is about 641 MB.

Table 1: Performance of VoiceGuard w.r.t. baseline kaldi and achieved word error rate (WER).

Test	WER	Baseline (s)	VoiceGuard (s)	Overhead
RM-bigram	2.3 %	351	522	48.5 %
RM-unigram	15.4 %	585	815	39.3 %
WSJ	18.1 %	1 427	2 854	100.1 %

We ran kaldi on an Intel Core i7-7700 CPU @ 3.6 GHz over every corpus and report the run time of each test in Tab. 1. The overhead of VoiceGuard is between 39 % and 49 % for RM and between 98 % and 104 % for WSJ. The higher overhead for WSJ is due to its larger model (graph) size: in the current version of SGX, enclaves can only use up to 96 MB memory and rely on swapping to access additional data. Even though processing time is doubled in some cases, our results show that VoiceGuard enables privacy-preserving speech processing even in real time.

4 OFFLINE MODEL GUARD (OMG) [3]

VoiceGuard [5], as presented in §3, depends on Intel SGX, a TEE implementation in Intel CPUs which are not commonly used in smartphones and tablets. However, there are usability issues when relying on online speech processing in mobile use cases: high latency and therefore a bad user experience occurs if the user has an unreliable or low-bandwidth network connection, and high roaming fees may apply if the user is abroad. In offline scenarios, cloud-based processing is impossible.

Therefore, we build Offline Model Guard (OMG) [3], a generic architecture that efficiently protects machine learning tasks on mobile devices and demonstrate its practicality using offline keyword recognition as an example application. OMG leverages SANCTUARY [6], a security architecture for the predominant mobile computing platform ARM. SANCTUARY mitigates existing flaws of ARM's own TEE implementation TrustZone via user-space enclaves: the speech processing tasks are executed in an environment that is protected via strict hardware-enforced two-way isolation from all other system components to minimize the attack surface.

Architecture Overview. The OMG architecture as visualized in Fig. 3 is very similar to the architecture of VoiceGuard described in §3. The main implementational difference is how SANCTUARY provides code and data integrity as well as data confidentiality while not negatively impacting the user experience: it leverages the ARM TrustZone Address Space Controller (TZASC) to exclusively bind attested memory to one (temporarily) dedicated CPU core running security-critical code on the Zircon microkernel [2], which provides a basic execution environment.

Note that SANCTUARY allows the user to directly and securely provide voice data to the enclave as it allows secure input from peripherals like the microphone. Conveniently, once the system is in the operation phase, it can be queried repetitively by the user, thereby avoiding repeated preparation and initialization costs as well as interaction with the vendor. To do this, after a query is processed, the SANCTUARY core can be reallocated to the commodity OS while the memory is still locked s.t. no device or core is able to access it. Then, when receiving a new query, a new SANCTUARY core is allocated and the locked memory is mapped to it in order to perform the processing task.

Side channel attacks that extract secrets from caches can be prevented easily since the L1 cache is core exclusive and the shared second level cache (L2) can be excluded from SANCTUARY memory without severe performance impact [6]. However, we assume that the adversary cannot perform hardware attacks, e.g., a physical side channel attack to extract secret keys.

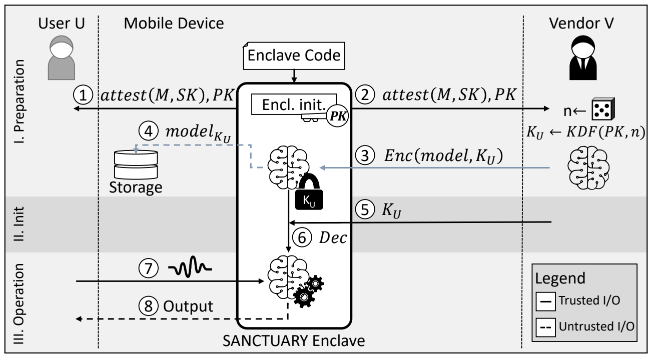


Figure 3: OMG architecture. The enclave is loaded and attested to user U and vendor V. V provides the encrypted ML model and sends the corresponding decryption key. U sends voice data to the enclave and receives respective textual output (which can be further processed into an action, as with virtual assistants).

Evaluation. We provide a fully functional prototype implementation of OMG on an ARM HiKey 960 development board for offline keyword recognition based on TensorFlow Lite for Microcontrollers [1]. The board is equipped with an ARMv8 octa-core SoC (4 cores @ 2.4 GHz, 4 cores @ 1.8 GHz) and 3 GB of RAM. We use such a development board instead of an off-the-shelf device since most vendors restrict developer access to TrustZone, which prevents us from setting up SANCTUARY. As our offline keyword recognition application is just a proof of concept, following [5], we do not focus on best accuracy, but study whether accuracy and runtime are affected when providing strong security guarantees.

The models are trained and evaluated on the Speech Command dataset [28] consisting of over 105,000 WAVE files of people saying 30 different words. The recordings were postprocessed to be a single word per file at a fixed 1 s duration. Features are computed using a 256 bin fixed point FFT across 30 ms windows (20 ms shift), averaging 6 neighboring bins, resulting in 43 values per frame. The 49 frames for each recording are concatenated, forming a fixed 49×43 compressed spectrogram ("fingerprint") per utterance.

The network architecture resembles [21], but is simplified to better match embedded requirements. It feeds the audio fingerprint to a 2D convolutional layer (8 filters, 8×10, x and y stride of 2), followed by ReLU activation and a regular layer that maps to the output labels. During training, dropout is applied after the convolution layer. We trained a system for a 12-class problem: *silence*, *unknown*, "yes", "no", "up", "down", "left", "right", "on", "off", "stop", "go". The model is first trained using TensorFlow and subsequently converted to a TensorFlow Lite and "micro" model. The resulting compressed model is 49 kB in size and achieves 75 % accuracy.

We evaluated the "micro" model on a subset of the published test set comprising 10 examples for each class, excluding the two rejection classes "silence" and "unknown", since sensitivity for those would typically be tuned for production. Running the test set on a 2.4 GHz core of the ARM development board takes 387 ms with and 379 ms without OMG protection, respectively. The runtimes are very close due to the fact that the hardware-enforced two-way isolation provided by SANCTUARY adds no additional overhead during execution. Since the duration of the test set is 100 s, the real-time factor is about 0.004.

In contrast to Intel SGX, SANCTUARY imposes no inherent memory limitations on our implementation, therefore it also allows to securely run more complex end-to-end systems, such as the very recently released and also TensorFlow-based offline dictation model by Google [22], making it highly practical.

ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of Education and Research (BMBF) and the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within the National Research Center for Applied Cybersecurity CRISP, by the DFG as part of HWSec, projects P3, S2, and E4 within the CRC 1119 CROSSING, and project A.1 within the RTG 2050 "Privacy and Trust for Mobile Users", by the Intel Collaborative Research Institute for Collaborative Autonomous & Resilient Systems (ICRI-CARS), the BioMobile II project (no. 518/16-30), the German Federal Office for Information Security within the research project DIRECT-PAD, and Omilia – Conversational Intelligence.

REFERENCES

[1] 2019. TensorFlow Lite for Microcontrollers. https://github.com/tensorflow/tensorflow/tree/master/tensorflow/lite/experimental/micro.

[2] 2019. Zircon Microkernel. https://fuchsia.googlesource.com/zircon.

[3] Sebastian P. Bayerl, Tommaso Frassetto, Patrick Jauernig, Korbinian Riedhammer, Ahmad-Reza Sadeghi, Thomas Schneider, Emmanuel Stapf, and Christian Weinert. 2020. Offline Model Guard: Secure and Private Speech Processing on Mobile Devices. In *DATE*. IEEE. To appear.

[4] Ferdinand Brasser, Srdjan Capkun, Alexandra Dmitrienko, Tommaso Frassetto, Kari Kostiainen, Urs Müller, and Ahmad-Reza Sadeghi. 2017. DR.SGX: Hardening SGX Enclaves against Cache Attacks with Data Location Randomization. *CoRR* abs/1709.09917 (2017).

[5] Ferdinand Brasser, Tommaso Frassetto, Korbinian Riedhammer, Ahmad-Reza Sadeghi, Thomas Schneider, and Christian Weinert. 2018. VoiceGuard: Secure and Private Speech Processing. In *INTERSPEECH*. ISCA.

[6] Ferdinand Brasser, David Gens, Patrick Jauernig, Ahmad-Reza Sadeghi, and Emmanuel Stapf. 2019. SANCTUARY: ARMing TrustZone with User-space Enclaves. In *Network & Distributed System Security Symposium (NDSS)*. Internet Society.

[7] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. 2018. The Secret Sharer: Measuring Unintended Neural Network Memorization & Extracting Secrets. *CoRR* abs/1802.08232 (2018).

[8] Sanchuan Chen, Xiaokuan Zhang, Michael K. Reiter, and Yinqian Zhang. 2017. Detecting Privileged Side-Channel Attacks in Shielded Execution with Déjà Vu. In *Asia Conference on Computer and Communications Security (ASIACCS)*. ACM.

[9] Anthony Cuthbertson. 2018. Amazon Ordered to Give Alexa Evidence in Double Murder Case. https://www.independent.co.uk/life-style/gadgets-and-tech/news/amazon-echo-alexa-evidence-murder-case-a8633551.html.

[10] Daniel Demmler, Thomas Schneider, and Michael Zohner. 2015. ABY-A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society.

[11] John Garofolo, David Graff, Doug Paul, and David Pallett. 2007. CSR-I,II (WSJ0,1) Complete. Linguistic Data Consortium, Philadelphia, USA.

[12] Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to Play any Mental Game. In *ACM Symposium on Theory of Computing (STOC)*. ACM.

[13] ISO/IEC JTC1 SC27 Security Techniques. 2011. *ISO/IEC 24745:2011. Information Technology - Security Techniques - Biometric Information Protection.* International Organization for Standardization.

[14] Seny Kamara and Mariana Raykova. 2011. Secure Outsourced Computation in a Multi-tenant Cloud. In *IBM Workshop on Cryptography and Security in Clouds*.

[15] Paul Kocher, Joshua Jaffe, and Benjamin Jun. 1999. Differential Power Analysis. In *Advances in Cryptology (CRYPTO)*. Springer.

[16] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. 1997. The DET Curve in Assessment of Detection Task Performance. In *European Conference on Speech Communication and Technology (EUROSPEECH)*. ISCA.

[17] National Institute of Standards and Technology (NIST). 2014. *The 2013-2014 Speaker Recognition i-vector Machine Learning Challenge.* Technical Report. NIST.

[18] Andreas Nautsch, Sergey Isadskiy, Jascha Kolberg, Marta. Gomez-Barrero, and Christoph Busch. 2018. Homomorphic Encryption for Speaker Recognition: Protection of Biometric Templates and Vendor Model Parameters. In *The Speaker and Language Recognition Workshop (Odyssey)*. ISCA.

[19] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. 2011. The Kaldi Speech Recognition Toolkit. In *Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE Signal Processing Society.

[20] P. Price, W. Fisher, Jared Bernstein, and David Pallett. 1993. Resource Management RM1 2.0. Linguistic Data Consortium, Philadelphia, USA.

[21] Tara Sainath and Carolina Parada. 2015. Convolutional Neural Networks for Small-Footprint Keyword Spotting. In *INTERSPEECH*. ISCA.

[22] Johan Schalkwyk. 2019. An All-Neural On-Device Speech Recognizer. https://ai.googleblog.com/2019/03/an-all-neural-on-device-speech.html.

[23] Hamza Shaban. 2018. Amazon Alexa User Receives 1,700 Audio Recordings of a Stranger through 'Human Error'. https://www.washingtonpost.com/technology/2018/12/20/amazon-alexa-user-receives-audio-recordings-stranger-through-human-error/.

[24] Ming-Wei Shih, Sangho Lee, Taesoo Kim, and Marcus Peinado. 2017. T-SGX: Eradicating Controlled-Channel Attacks Against Enclave Programs. In *Network & Distributed System Security Symposium (NDSS)*. Internet Society.

[25] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *Symposium on Security and Privacy (S&P)*. IEEE.

[26] Amos Treiber, Andreas Nautsch, Jascha Kolberg, Thomas Schneider, and Christoph Busch. 2019. Privacy-Preserving PLDA Speaker Verification using Outsourced Secure Computation. *Speech Communication* 114 (2019).

[27] Chia-che Tsai, Donald E. Porter, and Mona Vij. 2017. Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX. In *USENIX Annual Technical Conference (USENIX ATC)*. USENIX.

[28] Pete Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition.

[29] Andrew Chi-Chih Yao. 1986. How to Generate and Exchange Secrets. In *Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE.